RV College of Engineering

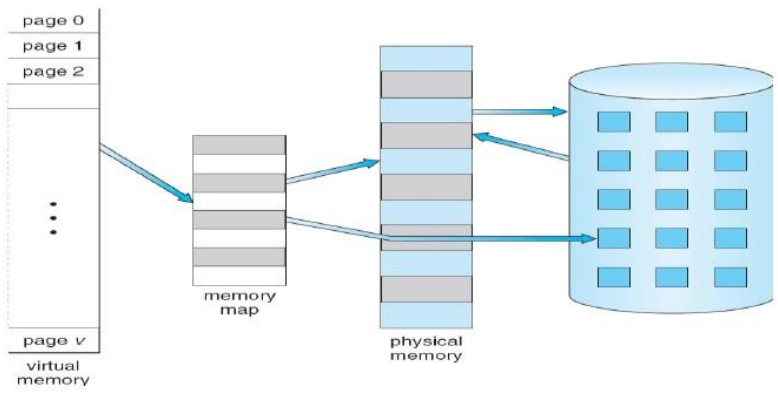
Go, change the world

Implementing Virtual Memory Management System

Name : ***Vikas J RVCE23BCS404***
Wilson RVCE23BCS418

Introduction to Implementing Virtual Memory Management System :

In the realm of computer science, the concept of virtual memory stands as a cornerstone for modern operating systems (OS). Virtual memory is a powerful abstraction that allows programs to operate as if they have access to a vast and contiguous block of memory, despite the limitations of physical RAM (Random Access Memory). virtual memory management plays a pivotal role in efficiently utilizing system resources and providing a seamless user experience. Operating Systems (OS) use virtual memory to abstract physical memory resources, enabling processes to access memory locations that may not necessarily reside in the main physical memory (RAM). This abstraction is crucial for multitasking environments where numerous processes compete for limited physical memory resources.

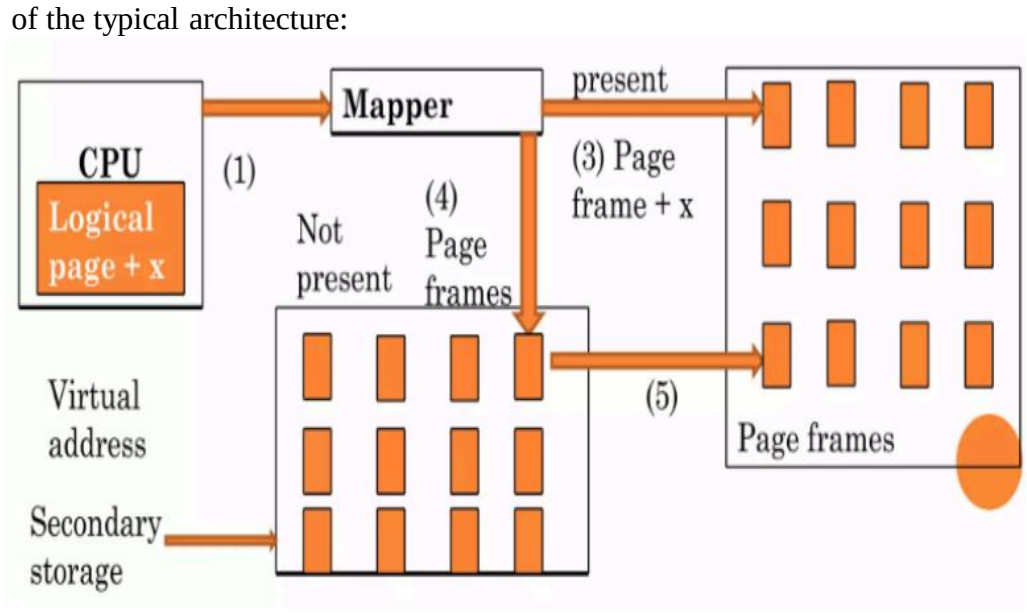


Problem Statement :

Design and implement an efficient virtual memory management system for an operating system to address the challenges of optimizing page replacement policies, minimizing page faults, and enhancing overall system performance. The system should be capable of dynamically managing the allocation and deallocation of virtual memory space, ensuring seamless interaction between physical and virtual memory, and implementing effective page replacement strategies.

System Architecture :

The system architecture of a virtual memory management system in an operating system involves several components working together to provide efficient memory abstraction, allocation, and management. Here's an overview of the typical architecture:



Methodology :

The methodology used in virtual memory management systems involves a combination of algorithms, techniques, and data structures to efficiently manage the virtual-to-physical memory mapping and handle memory operations.

Application :

1. VMM allows processes to access a larger address space than the physical memory available. Each process has its own virtual address space, which doesn't need to directly correspond to the physical RAM size.
2. Virtual memory enables memory protection by assigning different access permissions (read, write, execute) to different segments of memory. Unauthorized access attempts trigger memory protection faults, helping to enhance system security.
3. Virtual memory systems employ demand paging techniques to bring data into memory only when it is needed, reducing the amount of data that needs to be loaded into memory at any given time and improving overall system performance.

Source code

```python
#!/usr/bin/python

from utils import *
from vmmisc import *
import sys
import threading
import time
import random
import math
import os

memorySize = int(sys.argv[2])
pageSize = 1024
P = noPages = int(memorySize / pageSize)
B = noBitsForPage = int(math.log(P, 2))

process = {}
memory = [(-1, -1)] * P
readyqueue = []
blockedqueue = []
runningPID = -1
requests = {}
SIGUSR1 = SignalUser1()
memoryAccessEvent = threading.Event()

def getPageNo(vaddr):
    return int(hextobin(vaddr)[-16:-10], 2)

def scheduler():
    global runningPID
    while len(readyqueue) > 0:
        time.sleep(5)
        if len(readyqueue) > 0:  # Check if the ready queue is not empty
            runningPID = readyqueue.pop(0)  # Changed to pop from the beginning of the queue
            memoryAccessEvent.set()
```

Main.py

```python
from vmm import *
import sys, threading

f = open(sys.argv[1], 'r')
requestList = f.readlines()
f.close()

memorySize = int(sys.argv[2])
P = noPages = int(memorySize/pageSize)
B = noBitsForPage = int(math.log(P, 2))

for entry in requestList:
        pid, rw, vaddr = entry.split(',')
        pid = int(pid)
        rw = rw.strip()
        vaddr = vaddr.strip()
        if not process.__contains__(pid):
                processInit(pid)
        requests[pid].append((rw, vaddr))

thread_mmu = threading.Thread(target=mmu)
thread_os_scheduler = threading.Thread(target=scheduler)
thread_os_scheduler.start();
thread_mmu.start();
```

Output - 1

```
vikas@vikas-Inspiron-11-3162:~/Desktop/OS EL/Virtual Memory Manage
Scheduling.         pid:  1          vaddr:   F21B
Error: 'Frame Not Found. pid: 1, page: 60'
Error: Unable to perform memory access for pid: 1, vaddr: F21B
Blocked queue:      [1]
Loading.            pid:  1 , page:   60 , frame:   0
Scheduling.         pid:  2          vaddr:   A201
Error: 'Frame Not Found. pid: 2, page: 40'
Error: Unable to perform memory access for pid: 2, vaddr: A201
Blocked queue:      [1, 2]
Loading.            pid:  2 , page:   40 , frame:   0
Ready queue:        [3, 4, 5, 1]
Ready queue:        [3, 4, 5, 1, 2]
Scheduling.         pid:  3          vaddr:   4201
Error: 'Frame Not Found. pid: 3, page: 16'
Error: Unable to perform memory access for pid: 3, vaddr: 4201
Blocked queue:      [3]
Swapping.           pid:  2 , page:   40 , frame:   0
Scheduling.         pid:  4          vaddr:   8405
Error: 'Frame Not Found. pid: 4, page: 33'
Error: Unable to perform memory access for pid: 4, vaddr: 8405
Blocked queue:      [3, 4]
Loading.            pid:  4 , page:   33 , frame:   0
Loading.            pid:  3 , page:   16 , frame:   0
Ready queue:        [5, 1, 2, 3]
Ready queue:        [5, 1, 2, 3, 4]
Scheduling.         pid:  1          vaddr:   F24B
Direct Access.      024B

Direct Access.      024B

Main Memory:        | 3 |

Scheduling.         pid:  1          vaddr:   F21B
Direct Access.      021B

Direct Access.      021B
```

```
Scheduling.         pid:  1          vaddr:   F21B
Direct Access.      021B

Direct Access.      021B

Main Memory:        | 3 |

Scheduling.         pid:  3          vaddr:   3205
Direct Access.      0205

Direct Access.      0205

Main Memory:        | 3 |

Scheduling.         pid:  3          vaddr:   4201
Direct Access.      0201

Direct Access.      0201

Main Memory:        | 3 |

Scheduling.         pid:  4          vaddr:   8405
Direct Access.      0005

Direct Access.      0005

Main Memory:        | 3 |
```

Guide Information: Dr. Jyoti Shetty
Assistant Professor